APPLICATION FOR LETTERS PATENT

TO ALL WHOM IT MAY CONCERN:

BE IT KNOWN THAT **William Wai Lun Yip,** a citizen of Canada, having a residence at 85 Rowland Court Markham, Ontario L6C 1X8 Canada has invented a certain new and useful **TAMPER-PROOF MOBILE COMMERCE SYSTEM.**

TITLE OF INVENTION

TAMPER-PROOF MOBILE COMMERCE SYSTEM

FIELD OF INVENTION

This invention relates to debit transactions and more particularly a tamper-proof mobile commerce system utilizing a wireless device.

BACKGROUND OF THE INVENTION

As is well-known, credit or debit transactions are made through the utilization of credit cards or debit cards which are swiped through a reader at a register tip or cashier's terminal in order to authorize the payment of a purchase from a predetermined bank or financial institution.

Other cards with intelligence contained in the card, called smartcards, are often utilized at various ATM machines or other terminals to be able to withdraw cash or to effectuate some other debit transaction.

In the case of debit cards, key to the authorized use of such devices is the use of a so-called "PIN" which is a personal identification number that is known only to the individual seeking to cause the debit transaction to occur. For example, in a common debit card transaction, the merchant rings up the sale as usual, and the customer uses a specially provided piece of equipment to swipe his or her card, enter a PIN, and choose the bank from which the debit is to occur. The debit request is passed through, for example, either the Visa or MasterCard network, with the customer's card and PIN being transmitted to the bank where account funds are verified by the financial institution. Upon verification, the purchase is completed and funds are disbursed.

In supermarket checkouts, ATM machines, and in other places where PINs are often times required, casual observers can often times observe the PIN utilized by watching the individual key

in the PIN at a keypad at the terminal. The result is that debit cards can be effectively stolen without having possession of the debit card by obtaining the debit card number and associating it with a particular PIN.

Oftentimes what happens in stores is that unscrupulous store employees will, through access to the card reader or other devices, be able to draw down the individual's account or debit it through using the several pieces of information which are required, namely the account number, the person's name and the PIN.

The above is often called "shoulder surfing" and results in losses to the individual, calculated by the industry in the millions of dollars each year. While security systems are presumably in place to prevent the unauthorized use of a credit card or debit card, such systems can fail if those seeking to defeat the system either observe the payment card being used or have control of the equipment utilized to read the card.

As for credit cards, once the card is stolen or the account number and name is known, it is relatively easy to use the card or make purchases with the name, credit card number and expiration date.

Whether using a credit card or debit card, the payments are made in a process in which critical information as to the identity of the individual, and a personal ID number or an access PIN are utilized in authorizing the debit to be made from the bank or other financial institution.

More recently, wireless technologies have provided the ability for one having a wireless transmitter, such as a cellular phone, to be able to transact business, including surfing the internet, and can provide personal and banking information. One such example of a technology that enables such convenience is the Bluetooth™ protocol provided by the Bluetooth special interest group. It is essentially a cable replacement technology that uses a specific radio frequency range to allow Bluetooth-enabled devices to communicate wirelessly over short distances. People utilizing the

Bluetooth technology can utilize a wireless phone to, for instance, purchase soft drinks from a vending machine, pay parking meters, or, in fact, activate a juke box to play a particular song and have it charged to the individual's account.

With Bluetooth or similar technologies, there is also the potential to utilize the wireless technology to provide debiting of bank accounts in the electronic payment market.

While Bluetooth technology could enable users to complete debit transactions using their wireless phones, a disadvantage of the Bluetooth technology is that specialized hardware is required. In the vending machine scenario, a specialized receiver would have to be built into the vending machine, which is Bluetooth compatible. Moreover, the wireless devices used to achieve this transaction would need to contain a special Bluetooth chip. Thus, in a debit transaction, each cash register or debit terminal would need a Bluetooth receiver. While the Bluetooth-enabled phones could eliminate the risk of an observer observing a PIN, in order to build out such a Bluetooth system, much investment and long lead times are necessary.

The subject invention is a means of achieving debit and credit transactions that would have the security and convenience advantages of the technologies such as Bluetooth, but not its disadvantages.

SUMMARY OF THE INVENTION

In order to provide that a user can debit his or her bank account with complete security and without having the PIN being observable at the terminal at which the transaction takes place, in one embodiment the person seeking to debit the account is provided with a wireless transmitter and transitory transaction number which is provided from a mobile commerce server for each transaction and changes with each transaction. The user then keys this rather meaningless transitory number by using the keypad on the wireless transmitter, which identifies the particular

transaction and is good for no other transaction. Alternatively the transitory transaction number can be automatically inputted into the wireless transmitter at the transaction site using short-distance wireless transmission technology such as Bluetooth.

Thus, even if an observer observes the transaction number, since it is a one-time use only number, it will provide no particularly significant information to those seeking to provide further debit transactions.

Thus, in the subject invention, there is a change in the type of information that the user needs to enter as well as the equipment used to enter it. In conventional debit payment systems, shoppers enter their confidential bank information by swiping their bank card and entering their PIN into a terminal at the cash register where privacy cannot always be guaranteed. Because the terminal is in the possession of the store, the shopper has no way of knowing whether the terminal has been tampered with.

Thus, in the subject system the transaction is made secure because not only is no card involved, but when the shopper's sale is rung into the cash register, a unique transaction identification number is generated by the debit payment system and is displayed for the shopper. The shopper then enters the transaction identification number into his internet enabled cell phone in one embodiment.

The cell phone relays the transaction identification number along with the shopper's PIN information to the debit payment system. Not only is the previously sensitive transaction information now entered using the shopper's own equipment, but the nature of the information has changed. No longer is sensitive information provided by the user in terms of a PIN that can be observed. Rather, that which can be observed is only the transaction ID which is meaningless outside the context of the particular transaction. Note also that the PIN does not contain any bank information. Thus, even if the PIN is observed and even if the transaction ID is observed, the

person's account number is in no way available via the transaction. It is noted that the person's bank account number or card number is, of course, in the prior art observable either from looking at the credit card, tampering with a card reader, or having a copy of the credit card receipt.

Each user is uniquely identified to the mobile commerce system by a WAP ID or equivalent. It is noted that for wireless phones, each phone has an electronic serial number or ESN. The ESNs are not given out in Internet requests, but are instead translated into a unique WAP ID or equivalent in other non-WAP protocols.

The advantages of the above-noted mobile commerce solution are that it does not require special equipment and hardware thus eliminating the possibility of fraud through the equipment owned or operated by the store. The customer in this case has a trusted piece of equipment, namely his or her own mobile phone or wireless device, through which, by means of the WAP ID or equivalent, they are uniquely identified to the mobile commerce server. Moreover, because the mobile commerce solution requires only software modifications and, therefore, can leverage existing debit transaction equipment, stores can offer this method of paying by wireless device by keeping their existing equipment options. Additionally, a plastic debit/credit card is no longer required, eliminating the chance for loss, theft or fraud. Moreover, for debit cards users are no longer restricted to having to enter their PINs at the cash register. They can log into the service and enter the PIN in any part of the store where they have the desired privacy. Finally, the information that users enter is no longer sensitive in the same way as the payment card information. The transaction ID that is used is meaningless outside the transaction and has no value even if it were captured.

Note that the financial institution as used herein can be a bank, a credit or debit card company or even a store's own credit card facility.

In summary, a wireless phone or other wireless device is utilized to authorize debit transactions from a bank or financial institution in a secure manner in which a one-time meaningless transaction number is displayed to the user at the point of purchase terminal, which the user enters into his wireless phone or other wireless device to authorize the transaction. In one embodiment, the individual uses the device to call the mobile commerce server, and is identified by means of a Wireless Application Protocol ID, or equivalent. The user then enters a PIN number to authorize the transaction. The individual may also select from the wireless device the particular bank from which the debit is to come. In one embodiment, the mobile phone user then goes to the cash register and tells the clerk that this is a mobile commerce transaction. The transaction amount and the identity of the store is transmitted to the mobile commerce server, and the mobile commerce server transmits back to the register a one-time only transaction number which is displayed to the individual or automatically transmitted to the user's wireless device. The individual views the transaction number at the register and enters this number via the keypad into the wireless device if it has not already been automatically transmitted. The transaction number along with the PIN number or personal ID number and selected bank is then transmitted to the mobile commerce server, which authorizes and completes the sale, the fact of which is then transmitted back to the register. In so doing, casual observers will, if anything, obtain the transitory transaction number, which is meaningless. Moreover, any apparatus at the register, which would normally be utilized to transact the business, even if tampered with, would have no effect on the subject system since the apparatus, which initiates this transaction, is the wireless device, which is in the possession of the user as opposed to unscrupulous store employee or other miscreant.

BRIEF DESCRIPTION OF THE DRAWINGS

These and other features of the subject invention will be better understood in conjunction with the Detailed Description in connection with the Drawings of which:

Figure 1 is a diagrammatic representation of a scenario in which a user authorizes a debit through the utilization of a debit card and a PIN entry device, which is observable;

Figure 2 is a diagrammatic representation of the utilization of the subject system in which a transaction ID is transmitted back to the register terminal which is utilized by the customer to authorize the debit;

Figure 3 is a block diagram of the subject system illustrating the flow of events leading to an authorized debit;

Figure 4 is a front view of a wireless phone showing the phone display and an indication of what account is to be debited;

Figure 5 is a front view of the wireless phone of Figure 4, showing the entry of a PIN in masked form;

Figure 6 is a front view of the wireless phone of Figure 4, showing an on-screen prompt for the entry of a PIN;

Figure 7 is a front view of the wireless phone of Figure 4, showing the request to enter the transaction number;

Figure 8 is a front view of the wireless phone of Figure 4, the transaction ID number entered;

Figure 9 is a front view of this wireless phone of Figure 4, showing the amount of the payment to be authorized; and

Figure 10 is a front view of the phone of Figure 4 showing the screen indicating a completed transaction.

DETAILED DESCRIPTION

Referring now to Figure 1, while the subject system has application to both debit and credit card transactions, the debit card transaction is first described in this scenario, an individual can cause his or her debit card to be read by a card reader 12 at a register 14 attended by a clerk 16. In order for the desired transaction to be authorized, a PIN entry device 18 is provided at the register with the PIN being entered by individual 10. As can be seen, an observer 20 can see the individual entering the PIN and make mental note of it. Having the PIN and also some indication of the card number, the observer can then cause debits to be made on the individual's account. As mentioned hereinbefore, if the observer is a store employee or in collusion with a store employee, then obtaining the card number and marrying it with the PIN is a relatively easy task. An observer and an employee working in concert can therefore steal money from the accounts of many customers without the customer knowing that it is occurring.

It is also possible that a copy of the register receipt or credit card debit receipt can be obtained by one of the employees and correlated with the PIN that is observed.

While the above scenario is typical of an in-store debit card transaction, ATM transactions have essentially the same elements. Assuming that an inside employee can ascertain the debit card number, an observer can then observe the PINs being entered and correlate them with a particular card. Moreover, even if there is no inside employee at the ATM or in charge of the ATM, an observer can observe the debit card number from the debit card before it is inserted into the card slot.

While PIN-oriented security systems were designed to prevent against the above capture of the authorization information, it, nonetheless occurs indicating that the present PIN-oriented systems are not as secure as originally thought.

Referring now to Figure 2, in order to provide a secure transaction, an individual 30 utilizes a wireless phone 32 or other wireless device, which is connected via cell site 34 to a mobile commerce server 36. When the individual seeks to authorize a debit to his bank account at bank 38, a clerk 40 at a register 42 keys in the amount of purchase which is transmitted along with the store numbers illustrated by arrow 44 to mobile commerce server 36. The result is that the mobile commerce server transmits back a transaction ID number as illustrated by arrow 46 to register 42 where the transaction number is displayed at display 48 to the one seeking to authorize the debit.

Prior to the transaction, the mobile commerce server is made aware that the individual wishes to make a purchase by having the individual communicate with the mobile commerce server at some predetermined time before the actual transactions take place. At that time, the user transmits his PIN and the particular bank he wishes to use as the debiting authority. When the user now at the register tip sees the transaction ID number, he or she keys it into his wireless phone at which point it is transmitted to the mobile commerce server which then causes a debit transaction, here illustrated at 50 to occur at bank 38. Upon the correlation of the transaction number and the PIN as well as the bank, an authorization, here illustrated at 52, is sent back to register 42 indicating a completed transaction.

In this manner, the user is able to complete a debit transaction from his or her bank without having to use a debit card. Note that the only information necessary to be transmitted is the user's PIN and the bank or other financial institution from which the debit is to be made and that this is done in one embodiment prior to the individual arriving at the register.

In order to accomplish this transaction, the user knowing that he or she wishes to complete a transaction, calls up the mobile server via the wireless device and transmits the PIN and the bank identification to the mobile server. Thereafter, there is a timeout period in which the transaction must be made in order for the transaction to be authorized. For instance, a timeout period of five

minutes would not seem to be unreasonable to have the user alert the mobile commerce server that a transaction is coming and then go to the checkout counter and go through the checkout process.

In the subject invention, an electronic personal identification number is used to identify the individual to the mobile commerce server. In one embodiment this is a WAP ID. When a WAP enabled phone is used, the WAP ID is unique to the phone's ESN and can be used to identify the authorized individual each time the phone is used. In other non-WAP protocols, an equivalent ID is used.

Referring now to Figure 3, in general in one embodiment for debit cards a wireless phone 60 is utilized to communicate with mobile commerce server 36 through cell site 34 in which a PIN is entered on keypad 60. During the initial transaction in which the wireless phone communicates with the mobile commerce server, the user is identified to the mobile commerce server by means of a WAP ID or equivalent. After the user has authorized the transaction by entering his PIN, the mobile commerce server transmits back to the wireless phone those particular banking institutions, which are associated with the WAP ID or equivalent. Thereafter, the user specifies via keypad 60 that bank or financial institution, which is to be, utilized in the debit transaction, in this case bank 38. During the particular timeout period, the individual seeks to complete the transaction and the amount and store number as illustrated at 62 are transmitted to the mobile commerce server, whereas the aforementioned transaction number here illustrated at 64 is transmitted back to a register 66 where the transaction ID number is made available to the wireless phone user as illustrated at 68 then the transaction number is physically entered into the keypad or is automatically transmitted from the register to the wireless phone. Upon the transmission of the transaction ID number to the mobile commerce server, a debit is made from the user's bank account and the funds are transferred to the vendor here illustrated at 70.

In one scenario, the Mobile Commerce System provides a service to financial institutions or third party debit payment operators by offering this mode of payment. Member merchants of these financial institutions are debit payment operators who use the mobile commerce service automatically to gain the ability to offer payment by wireless phone to their customers. Customers may have multiple accounts with one or more member banks and as mentioned above, have the choice of paying for many of these accounts.

In one debit scenario, a customer has just finished shopping at, for instance, a supermarket. The customer takes out his cell phone and dials a pre-programmed mobile commerce website. The connection completes and the phone displays the accounts that the user can choose from. This is illustrated in Figure 4.

When the user decides to pay from a checking account, the user selects "CHK" and presses OK. Immediately, the user is prompted as shown in Figure 5 to enter a PIN for that account. For maximum security, the user is cautioned to use a quiet aisle in the supermarket to complete the login process and double check to make sure no one is close enough to watch the entry of the PIN.

Referring to Figure 6, as the PIN is entered, the display masks the PIN by displaying only an asterisk for each number that is entered. When the PIN has finally been entered, the user presses OK. The display then confirms which account has been chosen and prompts the user to enter a transaction number. At this point, the customer proceeds to the cash register knowing that he or she has five minutes before the authorization times out.

Referring to Figure 7, at the cash register, the register rings up the purchases and asks how the customer would like to pay. The customer responds "by mobile commerce" and the cashier punches a key on the debit payment terminal. After a few seconds, a transaction ID appears on the terminal screen which prompts the user to enter this number into the phone number and press OK. This entry is shown in Figure 8. As illuminated in Figure 9, the mobile server causes the phone to

display the transaction amount and asks for confirmation of payment. When the user presses OK, a final confirmation message appears indicating that the amount was paid.

The cash register then displays a similar message confirming that the transaction was completed successfully. Pressing "end" on the phone disconnects from the mobile commerce server at which point the customer can take the receipt and the purchases and leave the store.

In an alternative scenario for credit card purchases and as an alternative to current store credit cards, instead of providing credit cards which the customer swipes in card readers, participating stores can offer their customers access to their store accounts using their wireless phones. Assuming that one has finished shopping at, for instance, a discount chain store, one can take out one's cell phone and dial the store's website. The user's WAP ID or equivalent identifies him to the store's website. The connection completes and the user is prompted for the account's PIN. Choosing a quiet spot in the store, one makes sure that no one else is close enough to watch before the PIN is entered.

If PINs are used, once the PIN has been entered, the display masks the PIN by displaying only asterisks for each number that is entered. When having finished entering the PIN, the individual presses OK, at which point, the display prompts the entry of a transaction number. Heading for the cashier, one knows that he or she has five minutes before the authorization times out. Having proceeded to the cash register, the cashier rings up purchases and asks how the customer would like to pay. This is in essence a mobile commerce scenario described above with a message displaying the transaction amount and asking the individual to confirm payment which appears on the individual's phone where there is a screen capture of the amount confirmation.

When the customer presses OK, a message is displayed indicating that the amount was paid as displayed on-screen indicating that the transaction is complete. As before, there is a display at

the register of a similar message that the transaction has been completed. Pressing "End" on the wireless phone disconnects from the service.

What will be appreciated is that a cardless transaction has been completed which is secure and less prone to fraud than the use of either debit cards and associated PINs or credit cards with a handwritten signature.

This mobile commerce server can be utilized anywhere in which a debit is to be authorized from a financial institution whether or not it is in the form of a debit transaction or a credit card transaction. It can be used with current software and equipment normally found at registers or can be included in diverse devices where it is important that a PIN not be observable. Of course, not having a credit card masks the bank account and its owner from detection.

While the system is most readily adaptable at checkout counters and the like, this mobile commerce server can also be utilized with vending machines, parking meters, or other e-commerce transactions in which secure authorization is required. Thus, for instance, an individual's own computer could be used with increased security when performing an e-commerce transaction with one's own computer displaying the transaction number driven by a mobile commerce server, in this case, coupled to the internet.

This is because the identity of the bank and the individual is transmitted by another modality, namely, the wireless device. Thus, the individual's identity and bank are not available on the Internet as is the case with normal credit card transactions.

What is now presented is a program listing in Java, with the program to be run on WebLogic from BEA Systems:

DEBIT CARDS

ChkInput

```
Extern function set Variables()
{
      var user Input = WMLBrowser.getVar("combined");
```

```
	var payurl = WMLBrowser.getVar("payurl");

	var totLen = String.length(userInput);
	var part1 = String.subString(userInput, 0, 4);

	var len = totLen - 5;
	var remainder = String.subString(userInput, 5, len);

	var counter = 0;
	var zeroStr;
	while (String.charAt(remainder, counter) == 0) {
		if (counter == 0)
			zeroStr = "00";
		else
			zeroStr = zeroStr + "00";
		counter++;
	}

	if (counter > 0) {
		remainder = String.subString(remainder, counter, len-counter) +
zeroStr;
		len = String.length(remainder);
	}

	var centIndex = len - 2;
	var part2 = String.subString(remainder, 0, len - 2);
	var part3 = String.subString(remainder, centIndex, 2);
	var formattedPart2 = "$" + part2;

	if (counter == 0 && totLen < 8) {
		// not sufficient digits, send it back
		payurl = payurl + "#Correct";
	} else {
		WMLBrowser.setVar("transaction", part1);
		WMLBrowser.setVar("dollar", formattedPart2);
		WMLBrowser.setVar("cent", part3);
		payurl = payurl + "#Pay";
	}
	WMLBrowser.go(payurl);
}
```

Buyer1

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<!-- set document type to WML -->
<%@ page contentType="text/vnd.wap.wml" %>

<%@ page import="
javax.naming.*,
java.util.*,
java.sql.*,
weblogic.common.*
" %>
```

```
<%
      boolean ok = false;
      String myURL  = request.getRequestURI();
      int pos = myURL.lastIndexOf ("1.jsp");
      String nextURL;
      if (pos > 0) {
            nextURL = myURL.substring(0,pos) + ".jsp";
      } else {
            nextURL = "http://localhost:7001/interac/buyer.jsp";
      }

      String sub = null;
      String accType = null;
      String dollarStr = null;
      String centStr = null;
      String pin = null;
      String xactionStr = null;
      String combined = null;
      errorMsg = null;

      try {
            sub = request.getHeader("x-up-subno");
            accType = request.getParameter("acctype");
            dollarStr = request.getParameter("dollar");
            centStr = request.getParameter("cent");
            pin = request.getParameter("pin");
            xactionStr = request.getParameter("transaction");
            combined = request.getParameter("combined");
            ok = pay (sub, accType, dollarStr, centStr, pin, xactionStr,
combined);
      } catch (Exception e) { }

      String msg = "Transaction " + combined + " ($" + dollarStr + "." +
                        centStr + ") ";
      if (ok) {
            if (errorMsg != null)
                  msg = errorMsg;
            else
                  msg = msg + "done";
      } else {
            if (errorMsg != null)
                  msg = errorMsg;
            else
                  msg = msg + "rejected";
      }
%>


<wml>
      <head>
      <meta http-equiv="Cache-Control" content="max-age=0" forua="true"/>
      </head>
    <card  id="Done" ontimer="<%=nextURL%>">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="wait" value="50"/>
        </refresh>
    </onevent>
      <timer value="50" name="wait"/>
```

```
      <do  type="accept">
        <go href="<%=nextURL%>"/>
      </do>
      <p>
            <%=msg%>
      </p>
      </card>
</wml>

<%!
      String errorMsg = null;
      Connection conn  = null;
      Statement stmt = null;
      Statement stmt2 = null;
      ResultSet ds = null;
      ResultSet ds2 = null;
      String jdbcClass = "easysoft.sql.jobDriver";
      String jdbcURL   = "jdbc:easysoft://soft21ws:8831/interac2";

      private Connection getCon() {
            try {
                  Class.forName(jdbcClass).newInstance();
                  conn = DriverManager.getConnection(jdbcURL);
            } catch (Exception e) {}
            return conn;
      }

      private boolean pay (String sub, String accType,
                                    String dollarStr, String centStr,
                                    String pin, String xactionStr,
                                    String combined) {
            double reqAmt = (centStr.length() == 0) ? 0.0 :
Double.parseDouble(centStr)/100;
            reqAmt = reqAmt + Double.parseDouble(dollarStr.substring(1));
            try {
                  conn = getCon();
                  if (conn == null) {
                        errorMsg = "System not available.";
                        return false;
                  }
                  stmt = conn.createStatement();
            stmt.execute("select * from sales where xaction_id = " +
                                    xactionStr + " and sub_id = 0");
                  ds = stmt.getResultSet();

                  double amt = -0.01;
                  while (ds.next())  // should only be one
                        amt = ds.getDouble("amount");

                  if (amt == -0.01) {
                        errorMsg = "Invalid transaction " + combined + ".";
                        return false;
                  }

                  if (amt != reqAmt) {
                        errorMsg = "Invalid amt " + Double.toString(reqAmt) +
"."
                        return false;
                  }
```

```
                // next verify pin and balance
                stmt2 = conn.createStatement();
            stmt2.execute("select * from customer where sub = '" +
                                    sub + "' and account_type = '" + accType +
"'");
                ds2 = stmt2.getResultSet();

                String actualPin = "NOTPOSSIBLE";
                double balance = 0.0;
                long sub_id = 0;
                while (ds2.next()) { // should only be one
                      actualPin = ds2.getString("pin");
                      balance = ds2.getDouble("balance");
                      sub_id = ds2.getLong("sub_id");
                }

                if (!pin.equals(actualPin)) {
                      errorMsg = "Invalid pin.";
                      return false;
                }

                if (balance < reqAmt) {
                      errorMsg = "Insufficient balance " +
Double.toString(balance) + ".";
                      return false;
                }

                balance = balance - reqAmt;

                // Now confirmed user and transaction, update
                ds.close();
                ds2.close();
                stmt.close();
                stmt2.close();

                stmt = conn.createStatement();
            stmt.execute("update sales set sub_id = " +
                                    Long.toString(sub_id) +
                                    " where xaction_id = " +
                                    xactionStr + " and sub_id = 0");
                stmt2 = conn.createStatement();
            stmt2.execute("update customer set balance = " +
                                     Double.toString(balance) +
                                     " where sub = '" + sub +
                                     "' and account_type = '" + accType +
"'");
                return true;
            } catch (Exception e) { }
            finally {
                try {
                      ds.close();
                      ds2.close();
                      stmt.close();
                      stmt2.close();
                } catch (Exception e) { }
                try {
                      conn.close();
                } catch (Exception e) { conn = null; }
```

```
                conn = null;
            }
            return false;
        }
%>
```

Buyer

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<!-- set document type to WML -->
<%@ page contentType="text/vnd.wap.wml" %>

<%@ page import="
javax.naming.*,
java.util.*,
java.sql.*,
weblogic.common.*
" %>

<%
      String myURL  = request.getRequestURI();
      int pos = myURL.lastIndexOf (".jsp");
      String nextURL;
      if (pos > 0) {
            nextURL = myURL.substring(0,pos) + "1" + ".jsp";
      } else {
            nextURL = "http://localhost:7001/interac/buyer1.jsp";
      }

/*
      try {
            String sub = request.getHeader("x-up-subno");
            // determine the account types selectable based on subno
      } catch (Exception e) { }
*/

%>

<wml>
      <head>
      <meta http-equiv="Cache-Control" content="max-age=2592000" forua="true"/>
      </head>
    <card  id="Begin">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="acctype" value="CHQ"/>
            <setvar name="pin" value=""/>
            <setvar name="pintimeout" value="300"/>
        </refresh>
    </onevent>
      <do  type="accept">
          <go  href="#Xaction"/>
      </do>
      <p>
```

```
          Select account:
          <select name="acctype" ivalue="1">
                  <option value="CHQ">CHQ</option>
                  <option value="SAV">SAV</option>
            </select>
          Enter PIN:<input name="pin" type="password" format="NNNN"/>
<!--      Enter Timeout (secs):<input name="pintimeout" format="*N"/> -->
      </p>
    </card>

    <card  id="Xaction" ontimer="#Begin">
    <onevent type="onenterforward">
        <refresh>
              <setvar name="payurl" value="<%=myURL%>"/>
            <setvar name="combined" value=""/>
            <setvar name="transaction" value=""/>
            <setvar name="dollar" value=""/>
            <setvar name="cent" value=""/>
            <setvar name="timeout" value="$(pintimeout)0"/>
        </refresh>
    </onevent>
      <timer value="20" name="timeout"/>
      <do  type="accept">
        <go href="chkInput.wmls#setVariables()"/>
      </do>
      <p>
            Pay by $acctype <br/>
          Enter Transaction: <input name="combined" format="NNNN-NN*N"/>
      </p>
    </card>

    <card  id="Correct" ontimer="#Begin">
    <onevent type="onenterforward">
        <refresh>
              <setvar name="payurl" value="<%=myURL%>"/>
            <setvar name="transaction" value=""/>
            <setvar name="dollar" value=""/>
            <setvar name="cent" value=""/>
            <setvar name="timeout" value="300"/>
        </refresh>
    </onevent>
      <timer value="300" name="timeout"/>
      <do  type="accept">
        <go href="chkInput.wmls#setVariables()"/>
      </do>
      <p>
            Pay by $acctype <br/>
          Correct Transaction: <input name="combined" format="NNNN-NN*N"/>
      </p>
    </card>

    <card  id="Pay" ontimer="#Begin">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="timeout" value="50"/>
        </refresh>
    </onevent>
      <timer value="50" name="timeout"/>
      <do  type="accept" label="Yes">
```

```
        <go href="<%=nextURL%>">
            <postfield name="combined" value="$(combined)"/>
            <postfield name="transaction" value="$(transaction)"/>
            <postfield name="pin" value="$(pin)"/>
            <postfield name="dollar" value="$(dollar)"/>
            <postfield name="cent" value="$(cent)"/>
            <postfield name="acctype" value="$(acctype)"/>
            </go>
      </do>
      <do  type="options" label="No">
          <go  href="#Correct"/>
      </do>
      <p>
            Pay $(dollar).$(cent) ?
      </p>
    </card>
</wml>
```

Sales

```
<!doctype html public "-//w3c/dtd HTML 4.0//en">
<html>
<!-- Copyright (c) 1999-2000 by BEA Systems, Inc. All Rights Reserved.-->
<head>
<title>Mobile Commerce Demo</title>
</head>

<body bgcolor="#FFFFFF">
<font face="Helvetica">

<h2><font color=#DB1260>Cash Register Simulation</font></h2>

<hr width=80%>

<%@ page import="
javax.naming.*,
java.util.*,
java.sql.*,
weblogic.common.*
" %>

<%
      try {
            if (!"POST".equals(request.getMethod())) {    // first entry
                  String mode = request.getParameter ("mode");
                  if (mode == null || !mode.equals("wait")) {
%>
<p>
Enter store name and amount below.
</p>
<form method="post" name="Sales" action="Sales.jsp">

<table border=0 cellspacing=2 cellpadding=2 width=80%>

<tr>
<td width=30%><font face="Helvetica"><b>Storename :</b></td>
```

```
<td><font face="Helvetica"><input type="text" name="storename"
size=30></font></td>
</tr>

<tr>
<td width=30%><font face="Helvetica"><b>Amount :</b></td>
<td><font face="Helvetica"><input type="text" name="amount"
size=30></font></td>
</tr>

<tr>
<td><font face="Helvetica"><input type="Submit" value="Submit"
name="Submit"></td>
</tr>
</table>

</form>
<%
                  } else {
                        String storeName = request.getParameter ("storename");
                        String xaction = request.getParameter ("xaction");
                        boolean ok = waitForToken(storeName, xaction);
                        if (ok) {
%>
<p>
Amount paid.
</p>
<%
                        } else {
%>
<p>
Transaction failed.
<%= lastErrorMsg %>
</p>
<%
                        }
                  }

            } // first entry
            else {  // is a post
                  String storeName = request.getParameter ("storename");
                  String amount = request.getParameter ("amount");
                  conn = getCon();

                  int xactionId = random.nextInt(10000);
                  String xaction = cvtXaction (xactionId);
                  String xactionAmt = xaction + "-" + encodeAmount (amount);
                  if (conn != null) {
                        Statement stmt = conn.createStatement();
                        stmt.execute("insert into sales values
('"+storeName+"',"+
                                          xaction+",0,"+amount+")");
                  }

                  String myURL  = request.getRequestURI() +
"?mode=wait&storename=" +
                                    storeName + "&xaction=" + xaction;
                  String content = "1; url="+myURL;
                  response.setHeader("Refresh", content);
```

```
%>

<h2>Transaction ID: <%= xactionAmt%></h2>
<p>Waiting for payment in amount $ <%= amount%>.</p>

<%
            }  // is a post
%>

<%
      } catch (Exception e) {
%>

<p><b>There was a processing error: </b>
<br>
Exception: <%= e %>
<pre><%= getStackTraceAsString(e) %></pre>

<%
      }

%>

<p>
<font size=-1>Copyright (c) 2000 by Inc. All Rights Reserved.
</font>

<!--/font-->
</body>
</html>

<%!

      String getStackTraceAsString(Exception e)
      {
            // Dump the stack trace to a buffered stream, then return it's
contents
            // as a String. This is useful for printing the stack to 'out'.
            ByteArrayOutputStream ostr = new ByteArrayOutputStream();
            e.printStackTrace(new PrintStream(ostr));
            lastErrorMsg = ostr.toString();
            return(lastErrorMsg);
      }

      Connection conn  = null;
//    String jdbcClass = "COM.cloudscape.core.JDBCDriver";
//    String jdbcURL   = "jdbc:cloudscape:mobile";
      String jdbcClass = "easysoft.sql.jobDriver";
      String jdbcURL   = "jdbc:easysoft://soft21ws:8831/interac2";
      Random random    = new Random (System.currentTimeMillis());
      String lastErrorMsg = "";

      public Connection getCon() {
            try {
                  Class.forName(jdbcClass).newInstance();
                  conn = DriverManager.getConnection(jdbcURL);
            } catch (Exception e) { getStackTraceAsString(e); }
            return conn;
      }
```

```
	private boolean waitForToken(String store, String xaction) {
		long token = 0;
		Statement stmt = null;
		ResultSet ds = null;
		try {
			conn = getCon();
			if (conn == null)
				return false;
			while (true) {
				stmt = null;
				ds = null;
				stmt = conn.createStatement();
				stmt.execute("select * from sales where store_name='" +
store+
						"' and xaction_id = " + xaction);
				ds = stmt.getResultSet();
				while (ds.next())  // first one
					token = ds.getLong("sub_id");
				if (token != 0)
					break;

				ds.close();
				stmt.close();
				Thread.sleep(2000);
			}
		} catch (Exception e) {return false;}
		finally {
			try {
				ds.close();
				stmt.close();
			} catch (Exception e) { }
			try {
				conn.close();
			} catch (Exception e) { conn = null; }
			conn = null;
		}
		return true;
	}

	private String encodeAmount (String amtin) {
		String amt = amtin;
		int dot = amtin.lastIndexOf ('.');
		if (dot == -1 || dot == (amtin.length()-1)) {
			amt = amtin + "00";  // add cents
		} else if (dot == (amtin.length()-2)) {
			amt = amtin.substring (0, dot);
			amt = amt + amtin.substring (dot+1) + "0";
		} else {
			amt = amtin.substring (0, dot);
			amt = amt + amtin.substring (dot+1);
		}
		int count = 0;
		int len = amt.length();
		String zeros = null;
		while (amt.charAt(len-count-1) == '0') {
			++count;
			if ((count % 2) == 0)
				if (zeros == null)
```

```
                        zeros = "0";
                    else
                        zeros = zeros + "0";
            }

            if ((count % 2) == 1)
                --count;

            if (count == 0)
                return amt;

            String encode = amt.substring (0, len-count);
            return zeros + encode;
      }

      String cvtXaction (int xactionId) {
            String xaction = Integer.toString(xactionId);
            int l = xaction.length();
            if (l == 3)
                xaction = "0" + xaction;
            else if (l == 2)
                xaction = "00" + xaction;
            else if (l == 1)
                xaction = "000" + xaction;

            return xaction;
      }
%>
```

CREDIT CARDS

Start

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<!-- set document type to WML -->
<%@ page contentType="text/vnd.wap.wml" %>

<%@ page import="
javax.naming.*,
java.util.*,
java.sql.*,
weblogic.common.*
" %>

<!-- REVIEW SET REFERRER AND SECURITY -->

<%!
      private final String jdbcClass = "sun.jdbc.odbc.JdbcOdbcDriver";
      private final String jdbcURL = "jdbc:odbc:store";
      private String errorMsg = null;

      private Connection getCon() {
            Connection conn = null;
```

```
            try {
                  Class.forName(jdbcClass).newInstance();
                  conn = DriverManager.getConnection(jdbcURL);
            } catch (Exception e) {}
            return conn;
      }

      private boolean validateSub (String sub) {
            Statement stmt1 = null;
            ResultSet ds1 = null;
            Connection conn = null;

            try {
                  conn = getCon();
                  if (conn == null) {
                        errorMsg = "System not available.";
                        return false;
                  }
                  stmt1 = conn.createStatement();
            stmt1.execute("select * from subscribers where sub_id = '" + sub +
"'");
                  ds1 = stmt1.getResultSet();

                  if (ds1.next()) {
                        // validated user
                        return true;
                  }
            } catch (Exception e) { }
            finally {
                  try {
                        ds1.close();
                        stmt1.close();
                  } catch (Exception e) { }
                  try { conn.close(); } catch (Exception e) { }
            }
            return false;
      }
%>

<%
      String myURL  = request.getRequestURI();
      int pos = myURL.lastIndexOf ("start.jsp");
      String nextURL;
      if (pos > 0) {
            nextURL = myURL.substring(0,pos) + "shop.jsp";
      } else {
            nextURL = "http://localhost:7001/store/shop.jsp";
      }

      boolean oldSub = false;
      try {
            String sub = request.getHeader("x-up-subno");
            oldSub = validateSub (sub);
      } catch (Exception e) { }

      if (oldSub) {
%>

<wml>
```

```
      <head>
      <meta http-equiv="Cache-Control" content="max-age=1314000" forua="true"/>
      </head>
    <card id="Begin">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="pass" value=" "/>
        </refresh>
    </onevent>
    <onevent type="onenterbackward">
        <refresh>
            <setvar name="pass" value=" "/>
        </refresh>
    </onevent>
      <do type="accept">
        <go href="<%=nextURL%>">
            <postfield name="pass" value="$(pass)"/>
            </go>
      </do>
      <p>
            Welcome to Virtual Card Service! <br/>
          Enter password:<input name="pass" type="password" format="NNNN"/>
      </p>
    </card>
</wml>

<%
      } else {
%>

<wml>
      <head>
      <meta http-equiv="Cache-Control" content="max-age=0" forua="true"/>
      </head>
    <card id="First">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="pass1" value=""/>
        </refresh>
    </onevent>
    <onevent type="onenterbackward">
        <refresh>
            <setvar name="pass1" value=""/>
        </refresh>
    </onevent>
      <do type="accept">
        <go href="#Confirm"/>
      </do>
      <p>
            Welcome to Virtual Card Service! <br/>
            To enable your phone with the service, <br/>
          Select password:<input name="pass1" type="password" format="NNNN"/>
      </p>
    </card>

    <card id="Confirm">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="pass2" value=""/>
```

```
                    <setvar name="passurl" value="<%=myURL%>"/>
          </refresh>
      </onevent>
        <do type="accept">
          <go href="chkPass.wmls#chkPass()"/>
        </do>
        <p>
            Confirm password:<input name="pass1" type="password" format="NNNN"/>
        </p>
      </card>

      <card id="Confirmed">
        <do type="accept">
          <go href="<%=nextURL%>">
              <postfield name="pass" value="$(pass2)"/>
              </go>
        </do>
        <p>
            Proceed to register?
        </p>
      </card>

      <card id="Correct">
      <onevent type="onenterforward">
          <refresh>
              <setvar name="pass1" value=""/>
          </refresh>
      </onevent>
        <do type="accept">
          <go href="#Confirm"/>
        </do>
        <p>
              Passwords do not match. <br/>
            Select password again:<input name="pass1" type="password"
format="NNNN"/>
        </p>
      </card>

</wml>

<%
      }
%>
```

Shop

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">

<!-- set document type to WML -->
<%@ page contentType="text/vnd.wap.wml" %>

<%@ page import="
javax.naming.*,
java.util.*,
java.sql.*,
```

```
weblogic.common.*
" %>


<%!
      private final String jdbcClass = "sun.jdbc.odbc.JdbcOdbcDriver";
      private final String jdbcURL = "jdbc:odbc:store";
      private String errorMsg = null;

      private Connection getCon() {
            Connection conn = null;
            try {
                  Class.forName(jdbcClass).newInstance();
                  conn = DriverManager.getConnection(jdbcURL);
            } catch (Exception e) { }
            return conn;
      }

      private String[] getCard (Connection conn, String sub, String store) {
            Statement stmt1 = null;
            ResultSet ds1 = null;
            String [] cardDetails = new String[2];
            cardDetails[0] = "Not found";
            cardDetails[1] = "01/01";

            try {
                  stmt1 = conn.createStatement();
            stmt1.execute("select * from sub_cards where sub_id='" + sub +
                                    "' and store='" + store + "'");
                  ds1 = stmt1.getResultSet();

                  if (ds1.next()) {
                        cardDetails[0] = ds1.getString ("card_no");
                        cardDetails[1] = ds1.getString ("expiry");
                  }

            } catch (Exception e) { }
            finally {
                  try { stmt1.close(); ds1.close(); } catch (Exception e) { }
            }
            return cardDetails;
       }

       private boolean doCharge (String sub, String store, String amt, String
 xaction_id) {
            Statement stmt1 = null;
            Connection conn = null;

            try {
                  conn = getCon();
                  if (conn == null) {
                        errorMsg = "System not available.";
                        return false;
                  }
                  String [] cardDetails = getCard (conn, sub, store);
                  String cardNo = cardDetails[0];
                  String expiry = cardDetails[1];
                  stmt1 = conn.createStatement();
            stmt1.execute("insert into transactions values ('" + store +
```

```
                                        "','" + sub + "', now(), " + amt +
                                        ",'" + xaction_id + "','" + cardNo +
"','" +
                                        expiry + "')");
                  return true;
            } catch (Exception e) { }
            finally {
                  try { stmt1.close(); } catch (Exception e) { }
                  try { conn.close(); } catch (Exception e) { }
            }
            return false;
      }

      private boolean doAdmin (String sub, String store, String card, String
expiry, Vector list) {
            Statement stmt1 = null;
            Connection conn = null;

            try {
                  conn = getCon();
                  if (conn == null) {
                        errorMsg = "System not available.";
                        return false;
                  }
                  String [] cardDetails = getCard (conn, sub, store);
                  String cardNo = cardDetails[0];

                  if (!cardNo.equals("Not found"))
                        return false;

                  stmt1 = conn.createStatement();
            stmt1.execute("insert into sub_cards values ('" + sub +
                                    "','" + store + "','" + card + "','" +
                                    expiry + "')");
                  return true;
            } catch (Exception e) { }
            finally {
                  try { stmt1.close(); } catch (Exception e) { }
                  getStoreList (conn, sub, list);
                  try { conn.close(); } catch (Exception e) { }
            }
            return false;
      }

      private void getStoreList (Connection conn, String sub, Vector list) {
            Statement stmt1 = null;
            ResultSet ds1 = null;

            try {
                  stmt1 = conn.createStatement();
            stmt1.execute("select * from sub_cards where sub_id = '" + sub +
"'");
                  ds1 = stmt1.getResultSet();

                  int k = 0;
                  while (ds1.next()) {
                        list.ensureCapacity(list.capacity() + 1);
                        list.addElement (ds1.getString ("store"));
                  }
```

```
            } catch (Exception e) { }
            finally {
                  try { stmt1.close(); ds1.close(); } catch (Exception e) { }
            }
      }

      private boolean validateSub (String sub, String passwd, Vector list) {
            Statement stmt1 = null;
            ResultSet ds1 = null;
            Connection conn = null;

            try {
                  conn = getCon();
                  if (conn == null) {
                        errorMsg = "System not available.";
                        return false;
                  }
                  stmt1 = conn.createStatement();
            stmt1.execute("select * from subscribers where sub_id = '" + sub +
"'");
                  ds1 = stmt1.getResultSet();

                  if (ds1.next()) {
                        String thePass = ds1.getString ("password");
                        if (!passwd.equals(thePass))
                              return false;
                        // validated user, now must get his list of cards
                        getStoreList (conn, sub, list);
                  } else {
                        addSub (conn, sub, passwd);
                  }
                  return true;
            } catch (Exception e) { }
            finally {
                  try { ds1.close(); stmt1.close(); } catch (Exception e) { }
                  try { conn.close(); } catch (Exception e) { }
            }
            return false;
      }

      private void addSub (Connection conn, String sub, String passwd) {
            Statement stmt1 = null;

            try {
                  stmt1 = conn.createStatement();
            stmt1.execute("insert into subscribers values('" + sub + "','" +
                                    passwd + "')");

            } catch (Exception e) { }
            finally {
                  try { stmt1.close(); } catch (Exception e) { }
            }
      }
%>

<%
      String myURL  = request.getRequestURI();
      int pos = myURL.lastIndexOf ("shop.jsp");
      String nextURL;
```

```
    if (pos > 0) {
          nextURL = myURL.substring(0,pos) + "Start.jsp";
    } else {
          nextURL = "http://localhost:7001/store/Start.jsp";
    }

  String postAdm ="<go href=\"" + myURL + "\">\n" +
          "\t<postfield name=\"store\" value=\"$(store)\"/>\n" +
          "\t<postfield name=\"cardno\" value=\"$(cardno)\"/>\n" +
          "\t<postfield name=\"expiry\" value=\"$(expiry)\"/>\n" +
                "</go>";

    boolean isLogin = false;
    boolean loginGood = false;
    boolean isShopAttempt = false;
    boolean shopOK = false;
    boolean isAdministration = false;
    boolean hasBay = false;
    boolean hasCT = false;
    boolean hasSears = false;
    boolean hasZellers = false;
    String store=null;
    try {
          Vector cards = new Vector();
          String sub = request.getHeader("x-up-subno");
          String pass = request.getParameter("pass");
          store = request.getParameter("store");
          String xaction_id = request.getParameter("transaction");

          if (pass != null)  { // login attempt
                isLogin = true;
                loginGood = validateSub (sub, pass, cards);
          } else if (xaction_id != null) { // shop attempt
                isShopAttempt = true;
                String dollar = request.getParameter("dollar");
                String cent = request.getParameter("cent");
                String amt = dollar + "." + cent;
                shopOK = doCharge(sub, store, amt, xaction_id);
          } else { // must be administer, i.e. cardNo != null
                isAdministration = true;
                String cardNo = request.getParameter("cardno");
                String expiry = request.getParameter("expiry");
                doAdmin (sub, store, cardNo, expiry, cards);
          }

          for (int k = 0; k < cards.size(); ++k) {
                String cardk = (String)cards.elementAt(k);
                if (cardk.equals("Bay"))
                      hasBay = true;
                else if (cardk.equals("Cdn Tires"))
                      hasCT = true;
                else if (cardk.equals("Sears"))
                      hasSears = true;
                else if (cardk.equals("Zellers"))
                      hasZellers = true;
          }
    } catch (Exception e) { }
    // Done processing, start building output deck
%>
```

```
<%
      if (isLogin && !loginGood) {
            // login failed, go back to start page
%>
<wml>
      <head>
      <meta http-equiv="Cache-Control" content="max-age=0" forua="true"/>
      </head>

    <card id="Begin">
      <do  type="accept">
        <go href="<%=nextURL%>"/>
      </do>
      <p>
          Logged in.
      </p>
    </card>
</wml>

<%
            return;
      }

      if (isShopAttempt && shopOK) {
%>
<wml>
      <head>
      <meta http-equiv="Cache-Control" content="max-age=0" forua="true"/>
      </head>

    <card id="Begin" ontimer="<%=nextURL%>">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="timeout" value="50"/>
        </refresh>
    </onevent>
      <timer value="50" name="timeout"/>
      <do  type="accept">
        <go href="<%=nextURL%>"/>
      </do>
      <p>
          Charge request sent to <%=store%>.
      </p>
    </card>
</wml>

<%
            return;
      }

      if (isShopAttempt && !shopOK) {
%>

<wml>
      <head>
      <meta http-equiv="Cache-Control" content="max-age=0" forua="true"/>
      </head>
```

```
    <card id="Begin" ontimer="<%=nextURL%>">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="timeout" value="50"/>
        </refresh>
    </onevent>
      <timer value="50" name="timeout"/>
      <do  type="accept">
        <go href="<%=nextURL%>"/>
      </do>
      <p>
          Charge request failed.
      </p>
    </card>
</wml>

<%
            return;
      }
%>

<wml>
      <head>
      <meta http-equiv="Cache-Control" content="max-age=0" forua="true"/>
      </head>

    <card id="ShopBegin">
      <do type="accept" label=" ">
        <noop/>
      </do>
      <do type="options" label="Cancel">
        <go href="<%=nextURL%>"/>
      </do>
      <p>
            Select:<br/>
            <anchor title="Go"><go href="#ShopList"/>Shop</anchor><br/>
            <anchor title="Go"><go href="#AdminList"/>Manage cards</anchor>
      </p>
    </card>

    <card id="ShopList">
      <do type="accept" label=" ">
        <noop/>
      </do>
      <do type="options" label="Main">
        <go href="#ShopBegin"/>
      </do>
      <p>
            Select:<br/>
<% if (hasBay) { %>
            <anchor title="Go"><go href="#Bay"/>Bay</anchor><br/> <% } %>
<% if (hasCT) { %>
            <anchor title="Go"><go href="#CT"/>Canadian Tires</anchor><br/> <%
} %>
<% if (hasSears) { %>
            <anchor title="Go"><go href="#Sears"/>Sears</anchor><br/> <% } %>
<% if (hasZellers) { %>
            <anchor title="Go"><go href="#Zellers"/>Zellers</anchor><br/> <% }
%>
```

```
  </p>
</card>

<card id="GetAmt">
<onevent type="onenterforward">
    <refresh>
        <setvar name="dollar" value=""/>
    </refresh>
</onevent>
  <do type="accept">
    <go href="#GetCents"/>
  </do>
  <p>
        Enter Dollar: <input name="dollar" format="N*N"/>
  </p>
</card>

<card id="GetCents">
<onevent type="onenterforward">
    <refresh>
        <setvar name="cent" value=""/>
    </refresh>
</onevent>
  <do type="accept">
    <go href="#Confirm"/>
  </do>
  <p>
        Enter Cents: <input name="cent" format="NN"/>
  </p>
</card>

<card id="Confirm">
  <do type="accept" label="Yes">
    <go href="<%=myURL%>">
        <postfield name="dollar" value="$(dollar)"/>
        <postfield name="cent" value="$(cent)"/>
        <postfield name="transaction" value="$(transaction)"/>
        <postfield name="store" value="$(store)"/>
        <postfield name="session" value="dynamicValue"/>
        </go>
  </do>
  <do type="options" label="No">
    <go href="#ShopBegin"/>
  </do>
  <p>
        Charge $(dollar).$(cent) to $(store) card?
  </p>
</card>

<card id="Bay">
<onevent type="onenterforward">
    <refresh>
        <setvar name="store" value="Bay"/>
<% if (hasBay) {%> <setvar name="transaction" value=""/>
<% } else { %>    <setvar name="cardno" value=""/>
                      <setvar name="expiry" value=""/>
<% } %>
    </refresh>
</onevent>
```

```
      <do type="accept">
<% if (hasBay) {%>      <go href="#GetAmt"/>
<% } else { %>          <%=postAdm%> <% } %>
      </do>
      <p>
<% if (hasBay) {%> Enter Transaction: <input name="transaction" format="N-
NNN"/>
<% } else {%>     Enter Card Number: <input name="cardno" format="N-NNN-NNNN"/>
                        Enter Expiry (MM-YY): <input name="expiry" format="NN-
NN"/>
<% } %>
      </p>
    </card>

    <card id="CT">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="store" value="Cdn Tires"/>
<% if (hasCT) {%> <setvar name="transaction" value=""/>
<% } else { %>    <setvar name="cardno" value=""/>
                        <setvar name="expiry" value=""/>
<% } %>
        </refresh>
    </onevent>
      <do type="accept">
<% if (hasCT) {%> <go href="#GetAmt"/>
<% } else { %>          <%=postAdm%> <% } %>
      </do>
      <p>
<% if (hasCT) {%> Enter Transaction: <input name="transaction" format="NNN"/>
<% } else {%>     Enter Card Number: <input name="cardno" format="NNN NNN
NNNN"/>
                        Enter Expiry (MM-YY): <input name="expiry" format="NN-
NN"/>
<% } %>
      </p>
    </card>

    <card id="Sears">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="store" value="Sears"/>
<% if (hasSears) {%> <setvar name="transaction" value=""/>
<% } else { %>    <setvar name="cardno" value=""/>
                        <setvar name="expiry" value=""/>
<% } %>
        </refresh>
    </onevent>
      <do type="accept">
<% if (hasSears) {%>    <go href="#GetAmt"/>
<% } else { %>          <%=postAdm%> <% } %>
      </do>
      <p>
<% if (hasSears) {%> Enter Transaction: <input name="transaction" format="NN
NN"/>
<% } else {%>     Enter Card Number: <input name="cardno" format="NNN-NNN-
NNN"/>
                        Enter Expiry (MM-YY): <input name="expiry" format="NN-
NN"/>
```

```
<% } %>
      </p>
    </card>

    <card id="Zellers">
    <onevent type="onenterforward">
        <refresh>
            <setvar name="store" value="Zellers"/>
<% if (hasZellers) {%> <setvar name="transaction" value=""/>
<% } else { %>    <setvar name="cardno" value=""/>
                        <setvar name="expiry" value=""/>
<% } %>
        </refresh>
    </onevent>
      <do type="accept">
<% if (hasZellers) {%>  <go href="#GetAmt"/>
<% } else { %>          <%=postAdm%> <% } %>
      </do>
      <p>
<% if (hasZellers) {%> Enter Transaction: <input name="transaction" format="N-
NNN"/>
<% } else {%>     Enter Card Number: <input name="cardno" format="N-NNN-NNNN"/>
                        Enter Expiry (MM-YY): <input name="expiry" format="NN-
NN"/>
<% } %>
      </p>
    </card>

    <card id="AdminList">
      <do type="accept" label=" ">
        <noop/>
      </do>
      <do type="options" label="Main">
        <go href="#ShopBegin"/>
      </do>
      <p>
            Select:<br/>
<% if (!hasBay) { %>
            <anchor title="Go"><go href="#Bay"/>Bay</anchor><br/> <% } %>
<% if (!hasCT) { %>
            <anchor title="Go"><go href="#CT"/>Canadian Tires</anchor><br/> <%
} %>
<% if (!hasSears) { %>
            <anchor title="Go"><go href="#Sears"/>Sears</anchor><br/> <% } %>
<% if (!hasZellers) { %>
            <anchor title="Go"><go href="#Zellers"/>Zellers</anchor><br/> <% }
%>
      </p>
    </card>

</wml>
```

ChkPass

```
extern function chkPass()
{
      var pass1 = WMLBrowser.getVar("pass1");
```

```
        var pass2 = WMLBrowser.getVar("pass2");
        var passurl = WMLBrowser.getVar("passurl");

        if (pass1 == pass2) {
              // matching passwords, go to confirmed
              passurl = passurl + "#Confirmed";
        } else {
              passurl = passurl + "#Correct";
        }
        WMLBrowser.go(passurl);
}
```

Having now described a few embodiments of the invention, and some modifications and variations thereto, it should be apparent to those skilled in the art that the foregoing is merely illustrative and not limiting, having been presented by the way of example only. Numerous modifications and other embodiments are within the scope of one of ordinary skill in the art and are contemplated as falling within the scope of the invention as limited only by the appended claims and equivalents thereto.